SET - 1

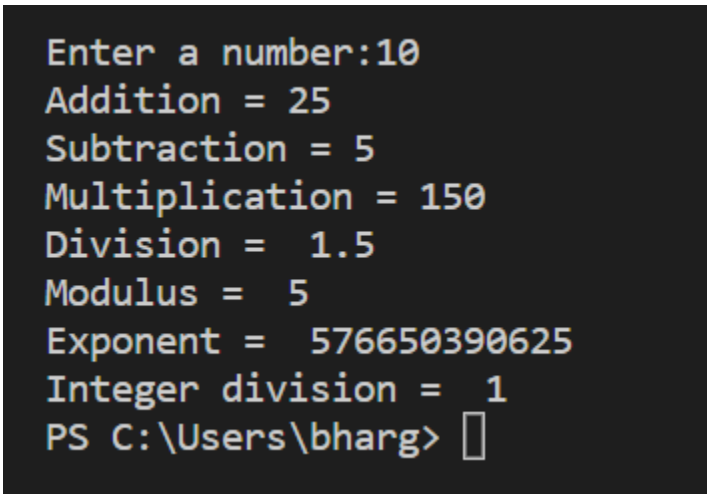
PRACTICAL NO: 1

Aim: Write a python program to create a simple arithmetic application including operations(addition, subtraction, multiplication, division, modulus, exponent, integer division.

Program :

```
a = int(input("Enter a number:"))
b = int(input("Enter a number:"))
print("Addition =" , (a + b))
print("Subtraction =" , (a - b))
print("Multiplication =" , (a * b))
print("Division = " , (a / b))
print("Modulus = " , (a % b))
print("Exponent = " , (a ** b))
print("Integer division = " , (a // b))
```

Output:



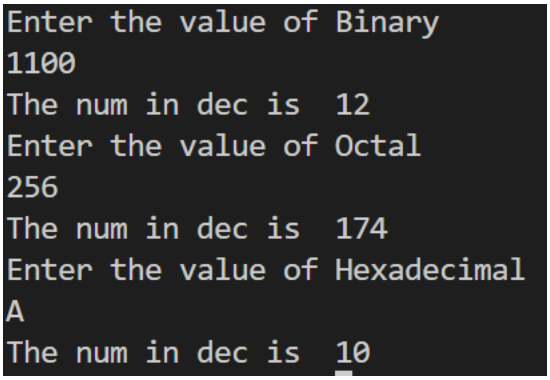
PRACTICAL NO: 2

Aim: Write a python program to convert numbers from octal, binary and hexadecimal systems into decimal number system.

Program:

```
a = input("Enter the value of Binary \n")
print("The num in dec is ",int(a,2))

a = input("Enter the value of Octal \n")
print("The num in dec is ",int(a,8))

a = input("Enter the value of Hexadecimal \n")
print("The num in dec is ",int(a,16))
```

Output:



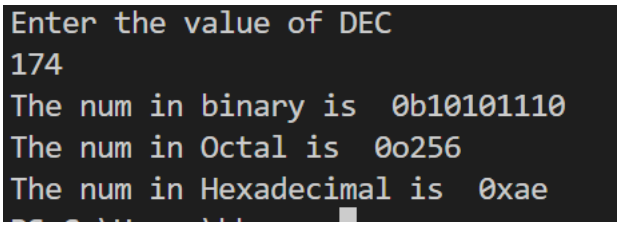
PRACTICAL NO: 3

Aim: Write a python program to convert numbers from decimal number system into octal, binary and hexadecimal system.

Program:

```
a =int( input("Enter the value of DEC \n"))
print("The num in binary is ",bin(a))

print("The num in Octal is ",oct(a))

print("The num in Hexadecimal is ",hex(a))
```

Output:



PRACTICAL NO: 4

Aim: Write a python program to check whether the given number is a palindrome or not.

Program:

```
no = int(input("Enter the Integer \n"))
sum = 0
temp = no
while(no>0):
    r=no%10
    sum=(sum*10)+r
    no = int(no / 10)
if(temp==sum):
    print(f"The num is palindrome {no}")
else:
    print("The numm is not palindrome !!!!!!")
```

Output:

```
Enter the Integer
121
The num is palindrome 0
```

PRACTICAL NO: 5

Aim: Write a python program to calculate area of a triangle.

Program:

```
x=float(input("Enter the val of Base "))
y=float(input("Enter the val of Height "))

Area =(x*y)/2
print("The area is",Area)
```

Output:

```
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\p4.py"
Enter the val of Base 6
Enter the val of Height 6
The area is 18.0
```

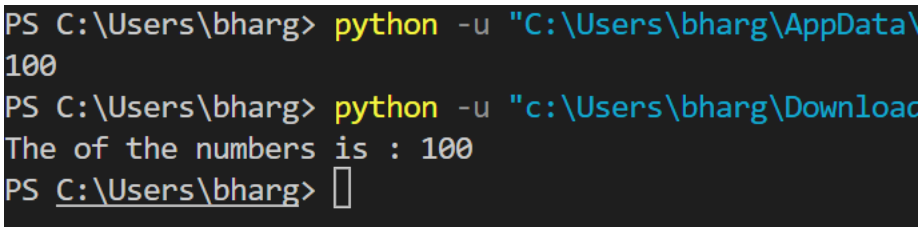
PRACTICAL NO: 6

Aim: Write a python program to display maximum of given 3 numbers.

Program:

```
x=3
y=7
z=100
if (x >= y) and (x >= z):

    largest = x

elif (y >= x) and (y >= z):

    largest = y

else:

    largest = z
print("The of the numbers is :" ,largest)
```

Output:



PRACTICAL NO: 7

Aim: Write a python program to find those numbers which are divisible by 3 and multiple of 5 within 500 numbers.

Program:

```
for x in range(1,501) :
   if(x % 3==0 and x % 5 == 0):
      print(x,end = ' ')
```

Output:

```
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\tempCodeRunnerFile.py"
15 30 45 60 75 90 105 120 135 150 165 180 195 210 225 240 255 270 285 300 315 330 345 360 375 3
90 405 420 435 450 465 480 495
PS C:\Users\bharg>
```

PRACTICAL NO: 8

Aim: Write a python program to draw kite pattern using for loop.

Program:

```
r=int(input("Enter the val for size "))
for x in range(r,0,-1):
    print(" " * x , "* " * (r-x))
for x in range(0,r,1):
    print(" " * x , "* " * (r-x))
for x in range(r-1,int(r/2)-1,-1):
    print(" " * x , "* " * (r-x))
```

Output:

```
PS C:\Users\bharg> python -u "C:\Users\bharg\AppData\Local\Temp\tempCodeRunnerFile.python"
100
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\Maxof3.py"
The of the numbers is : 100
     *
    * *
   * * *
  * * * *
 * * * * *
  * * * *
   * * *
    * *
     *
     *
    * *
   * * *
PS C:\Users\bharg>
```

SET - 2

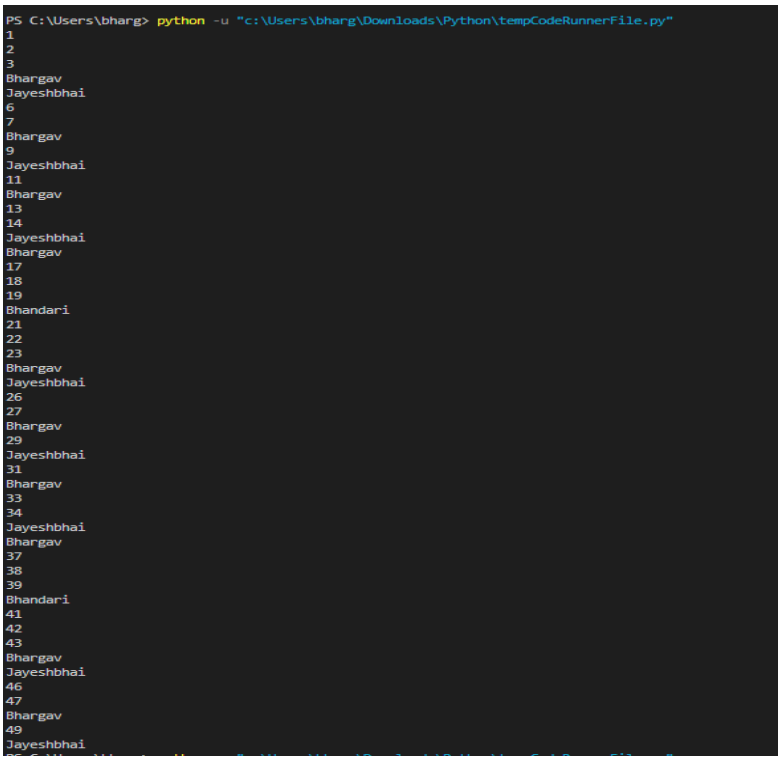
PRACTICAL NO: 1

Aim: Write a python program to print numbers from 1 to 50. For multiple of 4 print name instead of number and for multiple of 5 print father name. For the numbers which are multiple of both 4 and 5 print surname

Program:

```
for i in range(1,51):
    b=i
    if(i % 4 == 0):
        b="Bhargav"
    if(i % 5 == 0):
        b="Jayeshbhai"
    if(i % 4 == 0 and i % 5 ==0):
        b="Bhandari"
    print(b)
```

Output:



PRACTICAL NO: 2

Aim:

Write a python program to find numbers between 500 and 800 when each digit of number is ODD and the number should be printed in sequence separated by comma.

Program:

```
item=[]
for i in range(500,801):
   s=str(i)
   if(int(s[0])%2!=0) and (int(s[1])%2!=0) and (int(s[2])%2!=0):
      item.append(s)
print(", " .join(item))
```

Output:

```
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\tempCodeRunnerFile.py"
511, 513, 515, 517, 519, 531, 533, 535, 537, 539, 551, 553, 555, 557, 559, 571, 573, 575, 577, 579, 591, 593, 595, 597, 599, 711, 713,
715, 717, 719, 731, 733, 735, 737, 739, 751, 753, 755, 757, 759, 771, 773, 775, 777, 779, 791, 793, 795, 797, 799
PS C:\Users\bharg>
```

PRACTICAL NO: 3

Aim: Write a python program which accept a sequence of 4 digit binary numbers separated by comma and also print the numbers which are divisible by 3 in sequence separated by comma.

Program:

```
item=[]
print("Enter the  4 bit binary sequence sperated by comma ")
number = [x for  x in input().split(",") ]
for p in number:
    x=int(p,2)
    if (x % 3 ==0):
        item.append(p)
print(", " .join(item))
```

Output:

```
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\tempCodeRunnerFile.py"
Enter the  4 bit binary sequence sperated by comma
1110,1011,1111
1111
PS C:\Users\bharg>
```

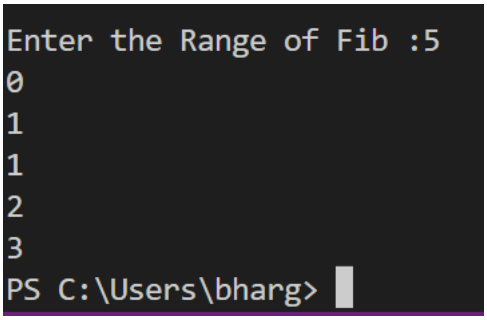
PRACTICAL NO: 4

Aim: Write a python program to display Fibonacci sequence up to nth term using recursive functions

Program:

```
def recurr_fib(n) :
    if n<=1:
        return n
    else :
        return(recurr_fib(n-1)+recurr_fib(n-2))
num = int(input("Enter the Range of Fib :"))

if num<=0:
    print("Enter a positive Number")
else :
    for i in range (num) :
        print(recurr_fib(i))
```

Output:



PRACTICAL NO: 5

Aim: Write a python program that accept a string and calculate the number of uppercase and lowercase letter.

Program:

```
str=input("Enter the string to check the lower and upper values \n")
b=0
m=0
for i in str:
    if (i<='z' and i>='a'):
        b=b+1
    if( i>='A' and i<='Z'):
        m=m+1

print("THE Lower CASE VLAUES ARE ",b)

print("THE Upper CASE VLAUES ARE ",m)
```

Output:

```
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\tempCodeRunnerFile.py"
Enter the string to check the lower and upper values
BhArGaV BHAndaRi
THE Lower CASE VLAUES ARE  7
THE Upper CASE VLAUES ARE  8
PS C:\Users\bharg>
```

PRACTICAL NO: 6

Aim: Write a python program to search a number in array using sequential search.

Program:

```
import numpy as nd
def seq_search(array,num) :
    pos=0
    found=False
    while pos<len(array) and not found:
        if (array[pos]==num):
            found=True
        else:
            pos=pos+1
    return found,pos+1
array1=nd.random.randint(50,size=(10))
print(array1)
number=int(input("Enter the number you want to search "))
print(seq_search(array1,number) )
```

Output:

```
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\2-p6.py"
[32 19 14 13 49 34 42 46  1 18]
Enter the number you want to search 56
(False, 11)
PS C:\Users\bharg>
```

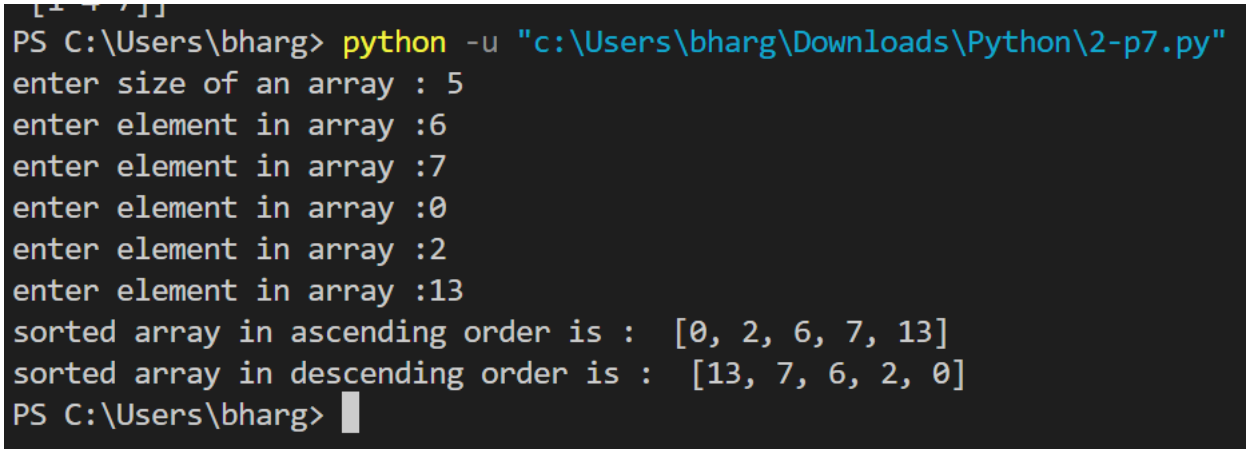
PRACTICAL NO: 7

Aim: Write a python program to sort elements of array.

Program:

```
size = int(input("enter size of an array : "))
arr = []
for i in range(size):
    element = int(input("enter element in array :"))
    arr.append(element)
    arr.sort()
print("sorted array in ascending order is : ",arr)
print("sorted array in descending order is : ",arr[::-1])
```

Output:



PRACTICAL NO: 8

Aim: Write a python program to input two matrix and perform the following given operation:

Program:

```
import numpy
arr1=([1,2,3],[4,5,6],[7,8,9])
arr2=([3,2,1],[6,5,4],[9,8,7])
result=([0,0,0],[0,0,0],[0,0,0])
mat1=numpy.array(arr1)
mat2=numpy.array(arr2)

print("The addition of matrix is \n",mat1+mat2)

print("The Subtraction of matrix is \n",mat1-mat2)

for i in range(0,len(mat1),1):
    for  k in range(0,len(mat2[0]),1):
        for j in range(0,len(mat2),1):
            result[i][j] +=mat1[i][k]*mat2[k][j]
print("The multiplication of matrix is ",)
for m in result:
    print(m)

print("The Transpose of matrix is \n",mat1.T)
print("The Transpose of matrix is \n",mat2.T)
```

Output:

```
PS C:\Users\bharg> python -u "c:\Users\bharg\Downloads\Python\2-p8.py"
The addition of matrix is
 [[ 4  4  4]
 [10 10 10]
 [2 5 8]
 [3 6 9]]
The Transpose of matrix is
 [[3 6 9]
 [2 5 8]
 [1 4 7]]
PS C:\Users\bharg>
```

SET – 3

PRACTICAL NO: 1

Aim: Do as directed: a) Read student data from given excel file sheet named as “5CSE” into appropriate pandas data structure. b) Fill missing values in columns “Subject” and “Batch” using forward fill method. c) Fill value “Jay Patel” in “Mentor” column for students having “Enrollment” column value from “200860131001” to “200860131029” and “Pal Patel” for remaining students. d) Add a new column “City” in existing student data and fill that column with residential city of student. e) Count total number of students subject-wise and batch-wise.

Program:

a)

```
import pandas as pd
df = pd.read_csv('5CSE.csv')
df
```

b)

```
df.drop("Unnamed: 0",axis=1,inplace=True)
df["Subject"].fillna(method="ffill",inplace=True)
df["Batch"].fillna(method="ffill",inplace=True)
df
```

c)

```
def  value(x):
    if(x<=200860131029):
        return "Jay Patel"
    else:
        return "Pal Patel"
df["Mentor"]=df.apply(lambda x:value(x["Enrollment"]),axis=1)
df
```

d)

```
df['City']=["Vapi","Silvassa","Vapi","Bhilad","Bhilad","Vapi","Vapi","Vapi","Vapi","Vapi","Vapi","Vapi","Vapi","Vapi","Silvassa","Vapi","Vapi","Vapi","Vapi","Udvada","Vapi","Vapi","Vapi","Vapi","Vapi","Vapi","Silvassa","Pardi","Vapi","Bhilad","Vapi","Vapi","Silvassa","Vapi","Vapi","Vapi","Vapi","Vapi","Vapi","Vapi","Bhilad","Vapi"]
df
```

e)

```
df.groupby(["Subject"]).size().reset_index(name='Number of Students')
df.groupby(["Batch"]).size().reset_index(name='Number of Students')
```

Output:

a)

	Unnamed: 0	Roll No	Enrollment	Name of Student	Subject	Batch	Mentor
0	0	1	200860131003	Arvindsingh Sarwansingh Deora	PDS	G1	NaN
1	1	2	200860131006	Sanal Saraswat	PDS	NaN	NaN
2	2	3	200860131007	Patel Yashkumar Hitendrabhai	PDS	NaN	NaN
3	3	4	200860131008	Harssht Shah	PDS	NaN	NaN
4	4	5	200860131010	Chaudhary Manisha K	PDS	NaN	NaN
5	5	6	200860131011	Nupur Kundan Bhavsar	PDS	NaN	NaN
6	6	7	200860131012	Kaustubh Vijay Tambe	PDS	NaN	NaN
7	7	8	200860131014	Devendra Ghori	PDS	NaN	NaN
8	8	9	200860131016	Harshalkumar Nalinbhai Pate	PDS	NaN	NaN
9	9	10	200860131017	Hadal Rinkesh Bhagvanbhai	PDS	NaN	NaN
10	10	11	200860131018	Patel Sneh	NaN	NaN	NaN

	Unnamed: 0	Roll No	Enrollment	Name of Student	Subject	Batch	Mentor
11	11	12	200860131019	Rutik Sumanbhai Patel	NaN	NaN	NaN
12	12	13	200860131020	Harshit.B.Dhodi	NaN	NaN	NaN
13	13	14	200860131021	Rahul Dora	NaN	G2	NaN
14	14	15	200860131022	Akshit Kantilal Patel	PDS	NaN	NaN
15	15	16	200860131023	Lakhani Brijay	PDS	NaN	NaN
16	16	17	200860131024	Krutagna Patel	PDS	NaN	NaN
17	17	18	200860131028	Dhruvi Sushilbhai Patel	PDS	NaN	NaN
18	18	19	200860131031	Mayur Pareshbhai Parmar	PDS	NaN	NaN
19	19	20	210860131503	Bhargav Bhandari	PDS	NaN	NaN
20	20	21	210860131504	Gautam Suraj Umaprasad	PDS	NaN	NaN
21	21	22	210860131506	Patel Prachi Jitendrabhai	PDS	NaN	NaN
22	22	23	210860131508	Ruchi P Mishra	PDS	NaN	NaN
23	23	24	210860131510	Hemangi Toke	PDS	NaN	NaN
24	24	25	210860131511	Sweta Vinod Jaiswal	PDS	NaN	NaN
25	25	26	210860131512	Siddhi Nagesh Bhad	PDS	NaN	NaN
26	26	27	200860131002	Amit Kumar Bhagat	CS	G3	NaN
27	27	28	200860131004	Harshi Patel	CS	NaN	NaN
28	28	29	200860131005	Mitaliya Vivek V.	CS	NaN	NaN
29	29	30	200860131009	Anuradha Kumari Kharwar	CS	NaN	NaN
30	30	31	200860131013	Yadav Kauntay Mahendra	NaN	NaN	NaN

	Unnamed: 0	Roll No	Enrollment	Name of Student	Subject	Batch	Mentor
31	31	32	200860131014	Saurabh Singh	NaN	NaN	NaN
32	32	33	200860131025	Hariom Prasad	CS	NaN	NaN
33	33	34	200860131026	Nisarg Ashokbhai Rathod	CS	NaN	NaN
34	34	35	200860131027	Rutik	CS	NaN	NaN
35	35	36	200860131029	Shivam Jaydeepbhai Desai	CS	NaN	NaN
36	36	37	200860131030	Rohan Prasad	CS	NaN	NaN
37	37	38	210860131501	Kaushik H Koladiya	CS	NaN	NaN
38	38	39	210860131502	Gohil Pratik D.	CS	NaN	NaN
39	39	40	210860131505	Tas Shaikh	CS	NaN	NaN
40	40	41	210860131509	Saloni Amar Bhatt	CS	NaN	NaN
41	41	42	210860131538	Nehal.M.Tandel	CS	NaN	NaN

b)

Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	
0	1	200860131003	Arvindsingh Sarwansingh Deora	PDS	G1	NaN
1	2	200860131006	Sanal Saraswat	PDS	G1	NaN
2	3	200860131007	Patel Yashkumar Hitendrabhai	PDS	G1	NaN
3	4	200860131008	Harssht Shah	PDS	G1	NaN
4	5	200860131010	Chaudhary Manisha K	PDS	G1	NaN
5	6	200860131011	Nupur Kundan Bhavsar	PDS	G1	NaN

Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	
6	7	200860131012	Kaustubh Vijay Tambe	PDS	G1	NaN
7	8	200860131014	Devendra Ghori	PDS	G1	NaN
8	9	200860131016	Harshalkumar Nalinbhai Pate	PDS	G1	NaN
9	10	200860131017	Hadal Rinkesh Bhagvanbhai	PDS	G1	NaN
10	11	200860131018	Patel Sneh	PDS	G1	NaN
11	12	200860131019	Rutik Sumanbhai Patel	PDS	G1	NaN
12	13	200860131020	Harshit.B.Dhodi	PDS	G1	NaN
13	14	200860131021	Rahul Dora	PDS	G2	NaN
14	15	200860131022	Akshit Kantilal Patel	PDS	G2	NaN
15	16	200860131023	Lakhani Brijay	PDS	G2	NaN
16	17	200860131024	Krutagna Patel	PDS	G2	NaN
17	18	200860131028	Dhruvi Sushilbhai Patel	PDS	G2	NaN
18	19	200860131031	Mayur Pareshbhai Parmar	PDS	G2	NaN
19	20	210860131503	Bhargav Bhandari	PDS	G2	NaN
20	21	210860131504	Gautam Suraj Umaprasad	PDS	G2	NaN
21	22	210860131506	Patel Prachi Jitendrabhai	PDS	G2	NaN
22	23	210860131508	Ruchi P Mishra	PDS	G2	NaN
23	24	210860131510	Hemangi Toke	PDS	G2	NaN
24	25	210860131511	Sweta Vinod Jaiswal	PDS	G2	NaN

Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	
25	26	210860131512	Siddhi Nagesh Bhad	PDS	G2	NaN
26	27	200860131002	Amit Kumar Bhagat	CS	G3	NaN
27	28	200860131004	Harshi Patel	CS	G3	NaN
28	29	200860131005	Mitaliya Vivek V.	CS	G3	NaN
29	30	200860131009	Anuradha Kumari Kharwar	CS	G3	NaN
30	31	200860131013	Yadav Kauntay Mahendra	CS	G3	NaN
31	32	200860131014	Saurabh Singh	CS	G3	NaN
32	33	200860131025	Hariom Prasad	CS	G3	NaN
33	34	200860131026	Nisarg Ashokbhai Rathod	CS	G3	NaN
34	35	200860131027	Rutik	CS	G3	NaN
35	36	200860131029	Shivam Jaydeepbhai Desai	CS	G3	NaN
36	37	200860131030	Rohan Prasad	CS	G3	NaN
37	38	210860131501	Kaushik H Koladiya	CS	G3	NaN
38	39	210860131502	Gohil Pratik D.	CS	G3	NaN
39	40	210860131505	Tas Shaikh	CS	G3	NaN
40	41	210860131509	Saloni Amar Bhatt	CS	G3	NaN
41	42	210860131538	Nehal.M.Tandel	CS	G3	NaN

c)

Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	
0	1	200860131003	Arvindsingh Sarwansingh Deora	PDS	G1	Jay Patel
1	2	200860131006	Sanal Saraswat	PDS	G1	Jay Patel
2	3	200860131007	Patel Yashkumar Hitendrabhai	PDS	G1	Jay Patel
3	4	200860131008	Harssht Shah	PDS	G1	Jay Patel
4	5	200860131010	Chaudhary Manisha K	PDS	G1	Jay Patel
5	6	200860131011	Nupur Kundan Bhavsar	PDS	G1	Jay Patel
6	7	200860131012	Kaustubh Vijay Tambe	PDS	G1	Jay Patel
7	8	200860131014	Devendra Ghori	PDS	G1	Jay Patel
8	9	200860131016	Harshalkumar Nalinbhai Pate	PDS	G1	Jay Patel
9	10	200860131017	Hadal Rinkesh Bhagvanbhai	PDS	G1	Jay Patel
10	11	200860131018	Patel Sneh	PDS	G1	Jay Patel
11	12	200860131019	Rutik Sumanbhai Patel	PDS	G1	Jay Patel
12	13	200860131020	Harshit.B.Dhodi	PDS	G1	Jay Patel
13	14	200860131021	Rahul Dora	PDS	G2	Jay Patel
14	15	200860131022	Akshit Kantilal Patel	PDS	G2	Jay Patel
15	16	200860131023	Lakhani Brijay	PDS	G2	Jay Patel
16	17	200860131024	Krutagna Patel	PDS	G2	Jay Patel
17	18	200860131028	Dhruvi Sushilbhai Patel	PDS	G2	Jay Patel

Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	
18	19	200860131031	Mayur Pareshbhai Parmar	PDS	G2	Pal Patel
19	20	210860131503	Bhargav Bhandari	PDS	G2	Pal Patel
20	21	210860131504	Gautam Suraj Umaprasad	PDS	G2	Pal Patel
21	22	210860131506	Patel Prachi Jitendrabhai	PDS	G2	Pal Patel
22	23	210860131508	Ruchi P Mishra	PDS	G2	Pal Patel
23	24	210860131510	Hemangi Toke	PDS	G2	Pal Patel
24	25	210860131511	Sweta Vinod Jaiswal	PDS	G2	Pal Patel
25	26	210860131512	Siddhi Nagesh Bhad	PDS	G2	Pal Patel
26	27	200860131002	Amit Kumar Bhagat	CS	G3	Jay Patel
27	28	200860131004	Harshi Patel	CS	G3	Jay Patel
28	29	200860131005	Mitaliya Vivek V.	CS	G3	Jay Patel
29	30	200860131009	Anuradha Kumari Kharwar	CS	G3	Jay Patel
30	31	200860131013	Yadav Kauntay Mahendra	CS	G3	Jay Patel
31	32	200860131014	Saurabh Singh	CS	G3	Jay Patel
32	33	200860131025	Hariom Prasad	CS	G3	Jay Patel
33	34	200860131026	Nisarg Ashokbhai Rathod	CS	G3	Jay Patel
34	35	200860131027	Rutik	CS	G3	Jay Patel
35	36	200860131029	Shivam Jaydeepbhai Desai	CS	G3	Jay Patel
36	37	200860131030	Rohan Prasad	CS	G3	Pal Patel

Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	
37	38	210860131501	Kaushik H Koladiya	CS	G3	Pal Patel
38	39	210860131502	Gohil Pratik D.	CS	G3	Pal Patel
39	40	210860131505	Tas Shaikh	CS	G3	Pal Patel
40	41	210860131509	Saloni Amar Bhatt	CS	G3	Pal Patel
41	42	210860131538	Nehal.M.Tandel	CS	G3	Pal Patel

d)

	Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	City
0	1	200860131003	Arvindsingh Sarwansingh Deora	PDS	G1	Jay Patel	Vapi
1	2	200860131006	Sanal Saraswat	PDS	G1	Jay Patel	Silvassa
2	3	200860131007	Patel Yashkumar Hitendrabhai	PDS	G1	Jay Patel	Vapi
3	4	200860131008	Harssht Shah	PDS	G1	Jay Patel	Bhilad
4	5	200860131010	Chaudhary Manisha K	PDS	G1	Jay Patel	Bhilad
5	6	200860131011	Nupur Kundan Bhavsar	PDS	G1	Jay Patel	Vapi
6	7	200860131012	Kaustubh Vijay Tambe	PDS	G1	Jay Patel	Vapi
7	8	200860131014	Devendra Ghori	PDS	G1	Jay Patel	Vapi
8	9	200860131016	Harshalkumar Nalinbhai Pate	PDS	G1	Jay Patel	Vapi
9	10	200860131017	Hadal Rinkesh Bhagvanbhai	PDS	G1	Jay Patel	Vapi
10	11	200860131018	Patel Sneh	PDS	G1	Jay Patel	Vapi
11	12	200860131019	Rutik Sumanbhai Patel	PDS	G1	Jay Patel	Vapi

	Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	City
12	13	200860131020	Harshit.B.Dhodi	PDS	G1	Jay Patel	Vapi
13	14	200860131021	Rahul Dora	PDS	G2	Jay Patel	Vapi
14	15	200860131022	Akshit Kantilal Patel	PDS	G2	Jay Patel	Silvassa
15	16	200860131023	Lakhani Brijay	PDS	G2	Jay Patel	Vapi
16	17	200860131024	Krutagna Patel	PDS	G2	Jay Patel	Vapi
17	18	200860131028	Dhruvi Sushilbhai Patel	PDS	G2	Jay Patel	Vapi
18	19	200860131031	Mayur Pareshbhai Parmar	PDS	G2	Pal Patel	Vapi
19	20	210860131503	Bhargav Bhandari	PDS	G2	Pal Patel	Udvada
20	21	210860131504	Gautam Suraj Umaprasad	PDS	G2	Pal Patel	Vapi
21	22	210860131506	Patel Prachi Jitendrabhai	PDS	G2	Pal Patel	Vapi
22	23	210860131508	Ruchi P Mishra	PDS	G2	Pal Patel	Vapi
23	24	210860131510	Hemangi Toke	PDS	G2	Pal Patel	Vapi
24	25	210860131511	Sweta Vinod Jaiswal	PDS	G2	Pal Patel	Vapi
25	26	210860131512	Siddhi Nagesh Bhad	PDS	G2	Pal Patel	Vapi
26	27	200860131002	Amit Kumar Bhagat	CS	G3	Jay Patel	Silvassa
27	28	200860131004	Harshi Patel	CS	G3	Jay Patel	Pardi
28	29	200860131005	Mitaliya Vivek V.	CS	G3	Jay Patel	Vapi
29	30	200860131009	Anuradha Kumari Kharwar	CS	G3	Jay Patel	Bhilad
30	31	200860131013	Yadav Kauntay Mahendra	CS	G3	Jay Patel	Vapi
31	32	200860131014	Saurabh Singh	CS	G3	Jay Patel	Vapi

	Roll No	Enrollment	Name of Student	Subject	Batch	Mentor	City
32	33	200860131025	Hariom Prasad	CS	G3	Jay Patel	Silvassa
33	34	200860131026	Nisarg Ashokbhai Rathod	CS	G3	Jay Patel	Vapi
34	35	200860131027	Rutik	CS	G3	Jay Patel	Vapi
35	36	200860131029	Shivam Jaydeepbhai Desai	CS	G3	Jay Patel	Vapi
36	37	200860131030	Rohan Prasad	CS	G3	Pal Patel	Vapi
37	38	210860131501	Kaushik H Koladiya	CS	G3	Pal Patel	Vapi
38	39	210860131502	Gohil Pratik D.	CS	G3	Pal Patel	Vapi
39	40	210860131505	Tas Shaikh	CS	G3	Pal Patel	Vapi
40	41	210860131509	Saloni Amar Bhatt	CS	G3	Pal Patel	Bhilad
41	42	210860131538	Nehal.M.Tandel	CS	G3	Pal Patel	Vapi

e)

	Batch	Number of Students
0	G1	13
1	G2	13
2	G3	16

	Subject	Number of Students
0	CS	16
1	PDS	26

PRACTICAL NO: 2

Aim: Do as directed: a) Read data from given csv file into appropriate pandas data structure. Delete rows having missing values. b) Calculate average price of cars having four and six cylinder engines. c) Find out cheapest and most expensive car details. d) Find out convertible and sedan car details having maximum engine horsepower. e) Find average sedan car price f) Count total number of cars per company. g) Find each company's highest car price.

Program:

a)

```
import pandas as pd
import numpy as np
df=pd.read_csv("Automobile_data_Miss.csv")
df
df.dropna(axis='rows')
```

b)

```
a1=df["num-of-doors"]=='four'
a_1=df[a1]["price"].mean()
a2=df["num-of-doors"]=='two'
a_2=df[a2]["price"].mean()
avg=(a_1+a_2)/2
print(avg)
```

c)

```
print("Detail of cheapest car")
df[df.price == df.price.min()]

print("Detail of expensive car")
df[df.price == df.price.max()]
```

d)

```
df1=df.loc[(df['body-style']=='convertible')]
df1=df1.iloc[df1['horsepower'].argmax()]

df2=df.loc[(df['body-style']=='sedan')]
df2=df2.iloc[df2['horsepower'].argmax()]
```

```
print('maximum hoursepower of convertible car is\n\n',df1)
print('\n\nmaximum hoursepower of sedan car is\n\n',df2)
```

e)

```
df[df["body-style"]=="sedan"].agg({"price":'mean'})
```

f)

```
df['make'].value_counts()
```

g)

```
car_manufacturers = df.groupby('make')
price= car_manufacturers["make","price"].max()
price
```

Output:

a)

```
import pandas as pd
import numpy as np
```

Python

```
df=pd.read_csv("Automobile_data_Miss.csv")
df
```

Python

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
0	3	NaN	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111.0	5000.0	21	27	13495.0
1	3	NaN	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111.0	5000.0	21	27	16500.0
2	1	NaN	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47	9.0	154.0	5000.0	19	26	16500.0
3	2	164.0	audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40	10.0	102.0	5500.0	24	30	13950.0
4	2	164.0	audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40	8.0	115.0	5500.0	18	22	17450.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
200	-1	95.0	volvo	gas	std	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114.0	5400.0	23	28	16845.0
201	-1	95.0	volvo	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	8.7	160.0	5300.0	19	25	19045.0
202	-1	95.0	volvo	gas	std	four	sedan	rwd	front	109.1	...	173	mpfi	3.58	2.87	8.8	134.0	5500.0	18	23	21485.0
203	-1	95.0	volvo	diesel	turbo	four	sedan	rwd	front	109.1	...	145	idi	3.01	3.40	23.0	106.0	4800.0	26	27	22470.0
204	-1	95.0	volvo	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114.0	5400.0	19	25	22625.0

205 rows × 26 columns

```
df.dropna(axis='rows')
```

Python

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
3	2	164.0	audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40	10.0	102.0	5500.0	24	30	13950.0
4	2	164.0	audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40	8.0	115.0	5500.0	18	22	17450.0
6	1	158.0	audi	gas	std	four	sedan	fwd	front	105.8	...	136	mpfi	3.19	3.40	8.5	110.0	5500.0	19	25	17710.0
8	1	158.0	audi	gas	turbo	four	sedan	fwd	front	105.8	...	131	mpfi	3.13	3.40	8.3	140.0	5500.0	17	20	23875.0
10	2	192.0	bmw	gas	std	two	sedan	rwd	front	101.2	...	108	mpfi	3.50	2.80	8.8	101.0	5800.0	23	29	16430.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
200	-1	95.0	volvo	gas	std	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114.0	5400.0	23	28	16845.0
201	-1	95.0	volvo	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	8.7	160.0	5300.0	19	25	19045.0
202	-1	95.0	volvo	gas	std	four	sedan	rwd	front	109.1	...	173	mpfi	3.58	2.87	8.8	134.0	5500.0	18	23	21485.0
203	-1	95.0	volvo	diesel	turbo	four	sedan	rwd	front	109.1	...	145	idi	3.01	3.40	23.0	106.0	4800.0	26	27	22470.0
204	-1	95.0	volvo	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114.0	5400.0	19	25	22625.0

160 rows × 26 columns

b)

```
13191.900236674213
```

c)

Detail of cheapest car

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
138	2	83.0	subaru	gas	std	two	hatchback	fwd	front	93.7	...	97	2bbl	3.62	2.36	9.0	69.0	4900.0	31	36	5118.0

1 rows × 26 columns

Detail of expensive car

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
74	1	NaN	mercedes-benz	gas	std	two	hardtop	rwd	front	112.0	...	304	mpfi	3.8	3.35	8.0	184.0	4500.0	14	16	45400.0

1 rows × 26 columns

d)

```
 symboling                3
normalized-losses       NaN
make                 porsche
fuel-type                gas
aspiration               std
num-of-doors             two
body-style       convertible
drive-wheels             rwd
engine-location         rear
wheel-base              89.5
length                 168.9
width                   65.0
height                  51.6
curb-weight             2800
engine-type             ohcf
num-of-cylinders         six
engine-size              194
fuel-system             mpfi
bore                    3.74
stroke                   2.9
compression-ratio        9.5
horsepower             207.0
peak-rpm              5900.0
...
city-mpg                  13
highway-mpg               17
price                36000.0
Name: 49, dtype: object
```

e)

```
price    14459.755319
dtype: float64
```

f)

```
toyota             32
nissan             18
mazda              17
mitsubishi         13
honda              13
volkswagen         12
subaru             12
peugot             11
volvo              11
dodge               9
mercedes-benz       8
bmw                 8
audi                7
plymouth            7
saab                6
porsche             5
isuzu               4
jaguar              3
chevrolet           3
alfa-romero         3
renault             2
mercury             1
Name: make, dtype: int64
```

g)

	make	price
make		
alfa-romero	alfa-romero	16500.0
audi	audi	23875.0
bmw	bmw	41315.0
chevrolet	chevrolet	6575.0
dodge	dodge	12964.0
honda	honda	12945.0
isuzu	isuzu	11048.0
jaguar	jaguar	36000.0
mazda	mazda	18344.0
mercedes-benz	mercedes-benz	45400.0
mercury	mercury	16503.0
mitsubishi	mitsubishi	14869.0
nissan	nissan	19699.0
peugot	peugot	18150.0
plymouth	plymouth	12764.0
porsche	porsche	37028.0
renault	renault	9895.0
saab	saab	18620.0
subaru	subaru	11694.0
toyota	toyota	17669.0
volkswagen	volkswagen	13845.0
volvo	volvo	22625.0

SET – 4

PRACTICAL NO: 1

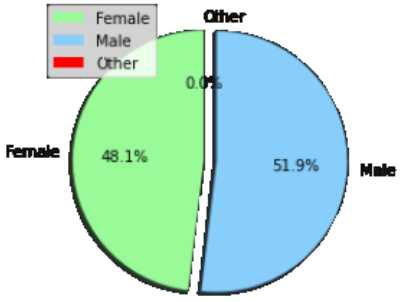
Aim: Plot gender-wise share of overall voters with legend and suitable labels. (Pie chart).

Program:

```
from pandas import DataFrame, read_csv
import matplotlib.pyplot as plt
import pandas as pd
result_df=pd.read_csv('Votes 2019.csv')
result_df
x=result_df["Female "].sum()
print("Sum of Female voters = ",x)
y=result_df["Male"].sum()
print("Sum of Male voters = ",y)
z=result_df["Other "].sum()
print("Sum of Other voters = ",z)

sizes=[x,y,z]
labels=["Female","Male","Other"]
colors=['palegreen','lightskyblue','red']
explode=(0,0.1,0)
plt.pie(sizes,explode=explode,labels=labels,colors=colors,startangle=90,autopct="%.1f%%",shadow=True)
plt.legend(labels,loc=2)
plt.show()
```

Output:

```
Sum of Female voters =  294035406
Sum of Male voters =  316792980
Sum of Other voters =  5663
```



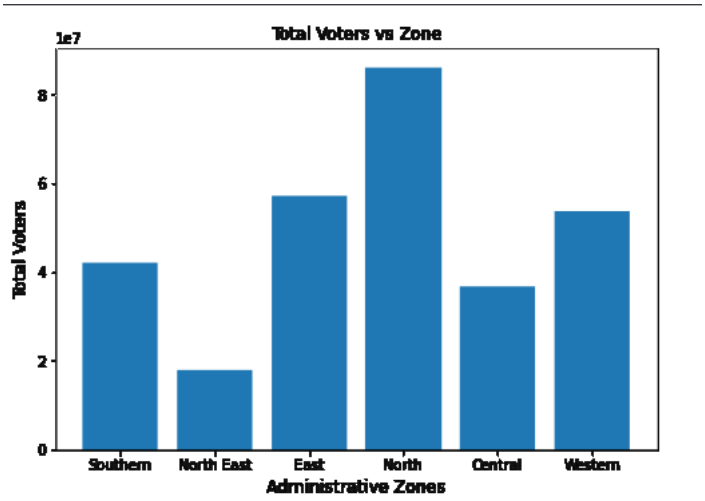
PRACTICAL NO: 2

Aim: Indian states are divided into six administrative zones: Central, East, North,Northeast, South and Western. Plot six bar chart into single figure to visualize total voters with suitable chart title.

Program:

```
zone=['Southern','Southern','North East','North
East','East','North','Central','Western','Western','Western','Western','North','North','North','East','South
ern','Southern','Southern','Central','Western','North','North East','North East','North
East','North','East','Southern','North','North','North East','Southern','Southern','North
East','North','North','East']
result_df['Zone']=zone
result_df.head()

result_df.to_csv("Votes_2019.csv")
y=result_df['Zone']
fig=plt.figure()
ax=fig.add_axes([0,3,1,1])
l1=ax.bar(y,result_df['Total Voters'])
ax.set_title("Total Voters vs Zone")
ax.set_ylabel("Total Voters",size=12)
ax.set_xlabel("Administrative Zones",size=12)
plt.show()
```

Output:



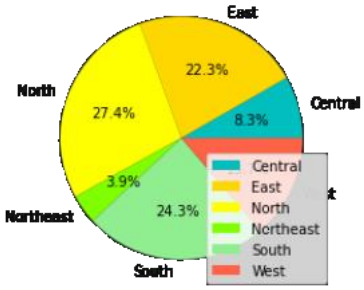
PRACTICAL NO: 3

Aim: Plot zone-wise share of total voters with legend and suitable labels. (Pie chart)

Program:

```
xx=result_df['Total Voters'].groupby(result_df['Zone']).sum()
print(xx)

sizes=[xx["Central"],xx["East"],xx["North"],xx["North East"],xx["Southern"],xx["Western"]]
labels=["Central","East","North","Northeast","South","West"]
colors=['c','gold','yellow','chartreuse','lightgreen','tomato']
plt.pie(sizes,colors=colors,labels=labels,autopct="%.1f%%")
plt.legend(labels,loc=4)

plt.show()
```

Output:



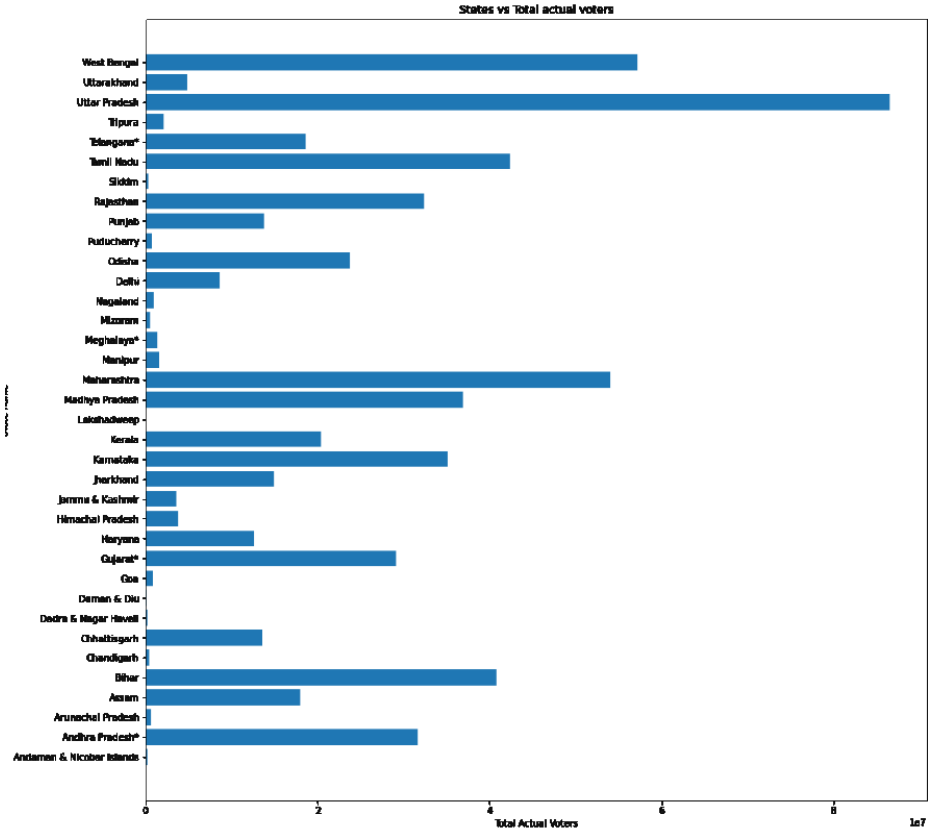
PRACTICAL NO: 4

Aim: Plot horizontal bar chart for states vs total actual votes with suitable labels.

Program:

```
fig=plt.figure()
ax=fig.add_axes([0,3,2,3])
ax.barh(y=result_df['State Name'],width=result_df['Total Actual Votes'])
plt.title("States vs Total actual voters")
plt.ylabel("State name")
plt.xlabel("Total Actual Voters")

plt.show()
```

Output:



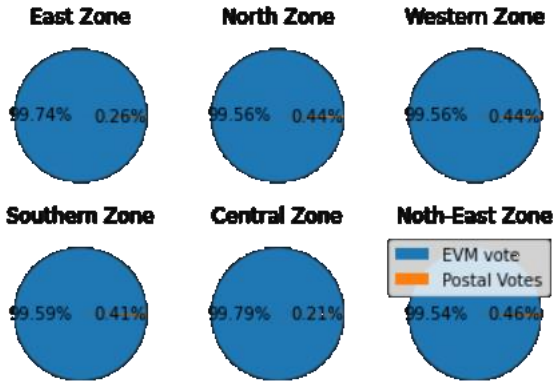
PRACTICAL NO: 5

Aim: Plot type-wise share (EVM and Postal) with legend and suitable labels for each administrative zone into single figure. (Pie chart).

Program:

```
import numpy as np
fig=plt.figure()
ax=fig.subplots(2,3)

xx=result_df['EVM Vote'].groupby(result_df['Zone']).sum()
yy=result_df['Postal Vote'].groupby(result_df['Zone']).sum()

ax[0][0].pie([xx['East'],yy['East']],autopct='%.2f%%')
ax[0][0].set_title('East Zone')
ax[0][1].pie([xx['North'],yy['North']],autopct='%.2f%%')
ax[0][1].set_title('North Zone')
ax[0][2].pie([xx['Western'],yy['Western']],autopct='%.2f%%')
ax[0][2].set_title('Western Zone')
ax[1][0].pie([xx['Southern'],yy['Southern']],autopct='%.2f%%')
ax[1][0].set_title('Southern Zone')
ax[1][1].pie([xx['Central'],yy['Central']],autopct='%.2f%%')
ax[1][1].set_title('Central Zone')
ax[1][2].pie([xx['North East'],yy['North East']],autopct='%.2f%%')
ax[1][2].set_title('Noth-East Zone')

plt.legend(["EVM vote","Postal Votes"])
plt.show()
```

Output:



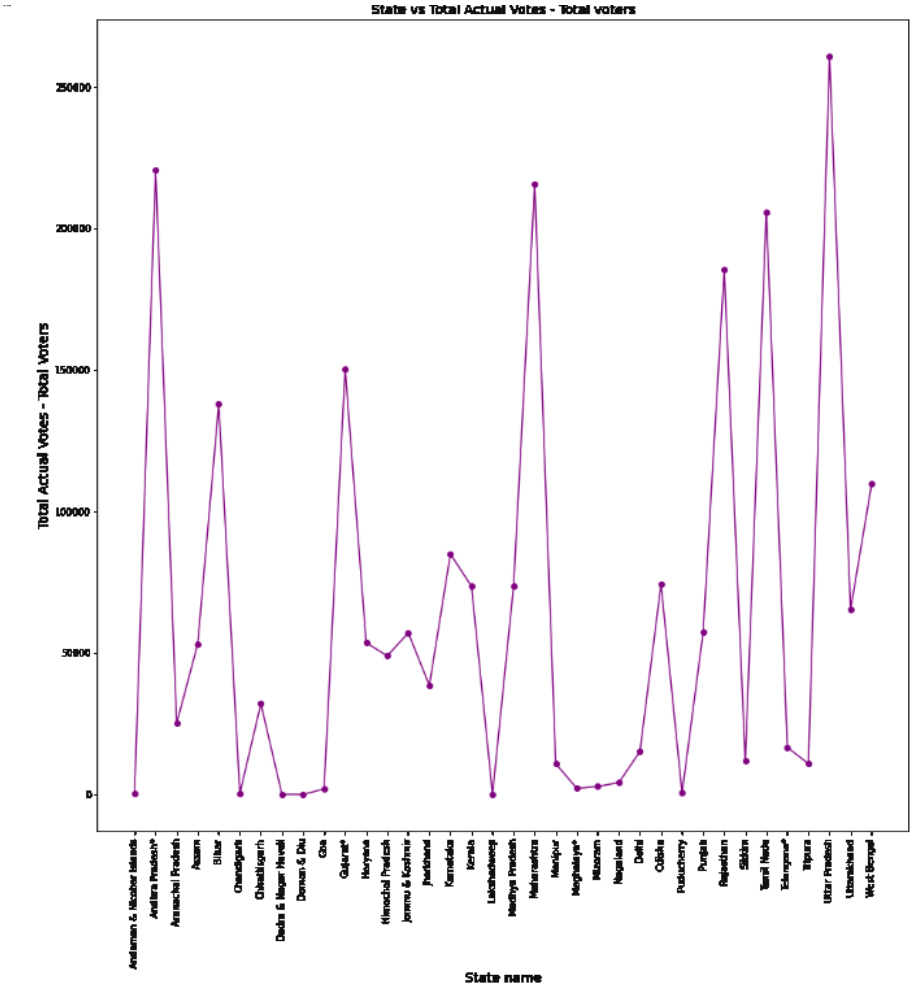
PRACTICAL NO: 6

Aim: Plot vote deficits (Total actual votes – Total voters) for each states using line chart

Program:

```
result_df.loc[:,"New"]=result_df.loc[:,"Total Actual Votes"].subtract(result_df.loc[:,"Total Voters"])
result_df
fig = plt.figure()
ax = fig.add_axes([0,3,2,3])
ax.plot(result_df['State Name'],result_df['New'],color='purple',marker='o')
plt.title("State vs Total Actual Votes - Total voters",fontsize=14)
plt.xticks(rotation=90)
plt.xlabel("State name",fontsize=14)
plt.ylabel("Total Actual Votes - Total Voters",fontsize=14)
plt.show()
```

Output:



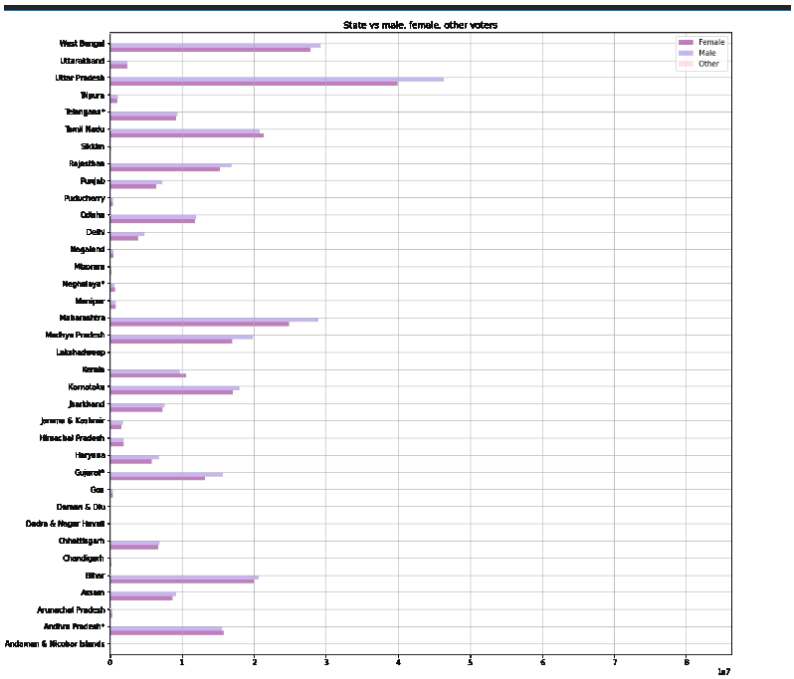
PRACTICAL NO: 7

Aim: Plot horizontal bar chart for states vs male, female and others votes (grouping of bars) with legend and suitable title.

Program:

```
fig = plt.figure()
ax = fig.add_axes([0,3,2,3])
pos=list(range(len(result_df["Female "])))
width=0.25
ax.barh(pos,result_df['Female '],width,alpha=0.5,color='purple',label=result_df['State Name'][0])
ax.barh([p + width for p in pos],
      result_df['Male'],width,alpha=0.5,color='mediumpurple',label=result_df['State Name'][1])
ax.barh([p + width*2 for p in pos],
      result_df['Other '],width,alpha=0.5,color='pink',label=result_df['State Name'][2])
ax.set_title('State vs male, female, other voters')
ax.set_yticks([p + 1.5*width for p in pos])
ax.set_yticklabels(result_df['State Name'])

plt.ylim(min(pos)-width,max(pos)+width*4)
plt.xlim([0,max(result_df['Female ']+result_df['Male']+result_df['Other '])])

plt.legend(['Female','Male','Other '],loc='upper right')
plt.grid()
plt.show()
```

Output:



SET - 5

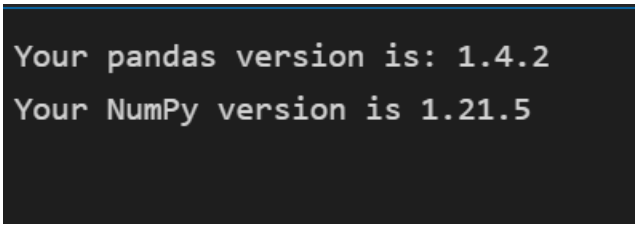
PRACTICAL NO: 1

Aim: Load iris dataset from sklearn library given iris.csv file into appropriate data structure of pandas.

Program:

```
%matplotlib inline
from sklearn.datasets import load_iris
iris = load_iris()

import pandas as pd
import numpy as np
print('Your pandas version is: %s' % pd.__version__)
print('Your NumPy version is %s' % np.__version__)
from sklearn.datasets import load_iris
iris = load_iris()
iris_nparray = iris.data

iris_dataframe = pd.DataFrame(iris.data, columns=iris.feature_names)
iris_dataframe['group'] = pd.Series([iris.target_names[k] for k in iris.target], dtype="category")
iris_dataframe
```

Output:



PRACTICAL NO: 2

Aim: Perform Descriptive Statistics for Numeric Data, Measuring central tendency, Measuring variance and range.

Program:

a) print(iris_dataframe.mean(numeric_only=True))

b) print(iris_dataframe.median(numeric_only=True))

c) print(iris_dataframe.std())

d) print(iris_dataframe.max(numeric_only=True) - iris_dataframe.min(numeric_only=True))

e) print(iris_dataframe.quantile([0,.25,.50,.75,1]))

Output:

a)

```
sepal length (cm)    5.843333
sepal width (cm)     3.057333
petal length (cm)    3.758000
petal width (cm)     1.199333
dtype: float64
```

b)

```
sepal length (cm)    5.80
sepal width (cm)     3.00
petal length (cm)    4.35
petal width (cm)     1.30
dtype: float64
```

c)

```
sepal length (cm)    0.828066
sepal width (cm)     0.435866
petal length (cm)    1.765298
petal width (cm)     0.762238
dtype: float64
```

d)

```
sepal length (cm)    3.6
sepal width (cm)     2.4
petal length (cm)    5.9
petal width (cm)     2.4
dtype: float64
```

e)

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0.00	4.3	2.0	1.00	0.1
0.25	5.1	2.8	1.60	0.3
0.50	5.8	3.0	4.35	1.3
0.75	6.4	3.3	5.10	1.8
1.00	7.9	4.4	6.90	2.5

+ Code + Markdown

PRACTICAL NO: 3

Aim: To Working with percentiles and defining measures of normality.

Program:

a)

```
from scipy.stats import kurtosis, kurtosistest
variable = iris_dataframe['petal length (cm)']
k = kurtosis(variable)
zscore, pvalue = kurtosistest(variable)
print('Kurtosis %0.3f z-score %0.3f p-value %0.3f' % (k, zscore, pvalue))
```

b)

```
from scipy.stats import skew, skewtest
variable = iris_dataframe['petal length (cm)']
s = skew(variable)
zscore, pvalue = skewtest(variable)
print('Skewness %0.3f z-score %0.3f p-value %0.3f' % (s, zscore, pvalue))
```

Output:

a)

```
Kurtosis -1.396 z-score -14.823 p-value 0.000
```

b)

```
Skewness -0.272 z-score -1.400 p-value 0.162
```

PRACTICAL NO: 4

Aim: To Counting for Categorical Data, Understanding frequencies, Creating contingency tables.

Program:

a)

```
pcts = [0, .25, .5, .75, 1]
iris_binned = pd.concat(
    [pd.qcut(iris_dataframe.iloc[:,0], pcts, precision=1),
     pd.qcut(iris_dataframe.iloc[:,1], pcts, precision=1),
     pd.qcut(iris_dataframe.iloc[:,2], pcts, precision=1),
     pd.qcut(iris_dataframe.iloc[:,3], pcts, precision=1)],
    join='outer', axis = 1)
print(iris_dataframe['group'].value_counts())
```

b)

```
print(iris_binned['petal length (cm)'].value_counts())
```

c)

```
print(pd.crosstab(iris_dataframe['group'], iris_binned['petal length (cm)']))
```

Output:

a)

```
setosa        50
versicolor    50
virginica     50
Name: group, dtype: int64
```

b)

```
(0.9, 1.6]    44
(4.4, 5.1]    41
(5.1, 6.9]    34
(1.6, 4.4]    31
Name: petal length (cm), dtype: int64
```

c)

petal length (cm)	(0.9, 1.6]	(1.6, 4.4]	(4.4, 5.1]	(5.1, 6.9]
group				
setosa	44	6	0	0
versicolor	0	25	25	0
virginica	0	0	16	34

PRACTICAL NO: 5

Aim: To Creating Applied Visualization for EDA like boxplots.

Program:

a)

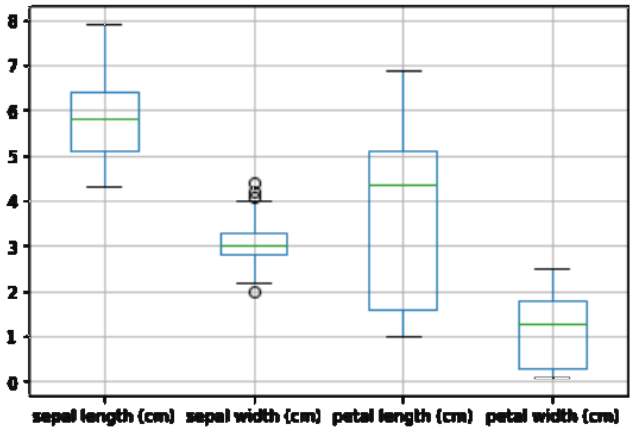
```
boxplots = iris_dataframe.boxplot(fontsize=9)
```

b)

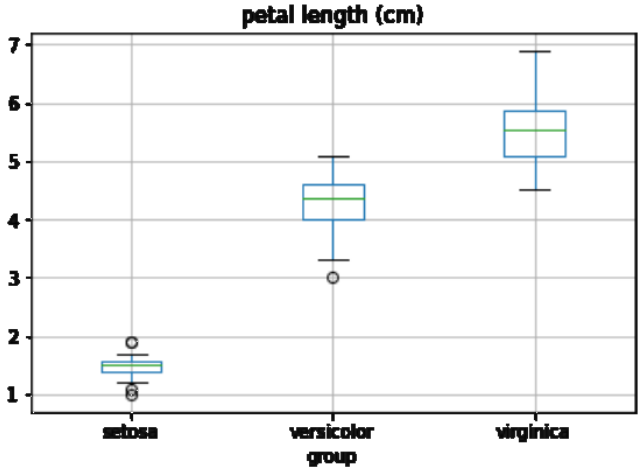
```
import matplotlib.pyplot as plt
boxplots = iris_dataframe.boxplot(column='petal length (cm)', by='group', fontsize=10)
plt.suptitle("")
plt.show()
```

Output:

a)



b)



SET - 6

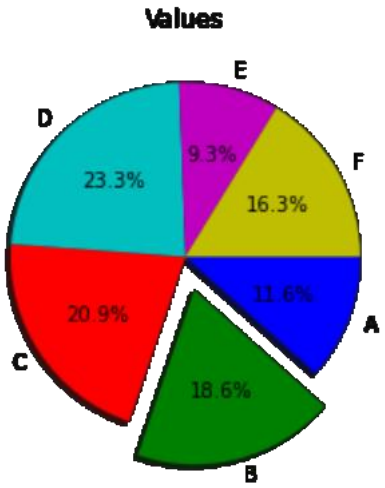
PRACTICAL NO: 1

Aim: Prepare a Pie charts by taking suitable reference.

Program:

```
import matplotlib.pyplot as plt
%matplotlib inline

values = [5, 8, 9, 10, 4, 7]
colors = ['b', 'g', 'r', 'c', 'm', 'y']
labels = ['A', 'B', 'C', 'D', 'E', 'F']
explode = (0, 0.2, 0, 0, 0, 0)

plt.pie(values, colors=colors, labels=labels, explode=explode, autopct='%1.1f%%',
counterclock=False, shadow=True)
plt.title('Values')

plt.show()
```

Output:



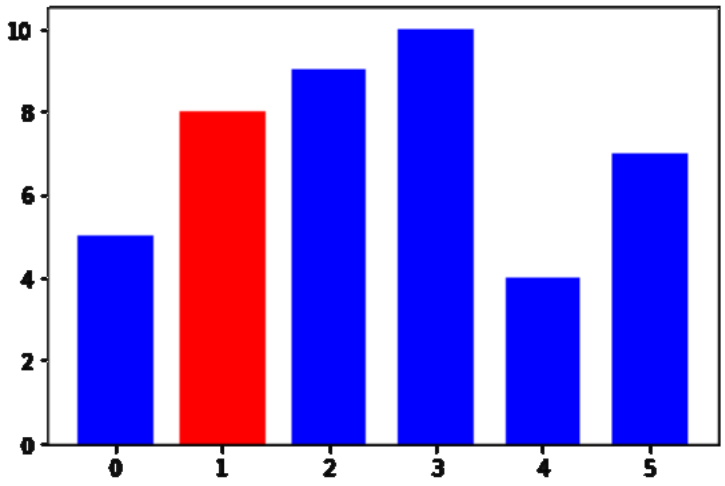
PRACTICAL NO: 2

Aim: Prepare a Barcharts charts by taking suitable reference.

Program:

```
import matplotlib.pyplot as plt
%matplotlib inline

values = [5, 8, 9, 10, 4, 7]
widths = [0.7, 0.8, 0.7, 0.7, 0.7, 0.7]
colors = ['b', 'r', 'b', 'b', 'b', 'b']
plt.bar(range(0, 6), values, width=widths, color=colors, align='center')

plt.show()
```

Output:



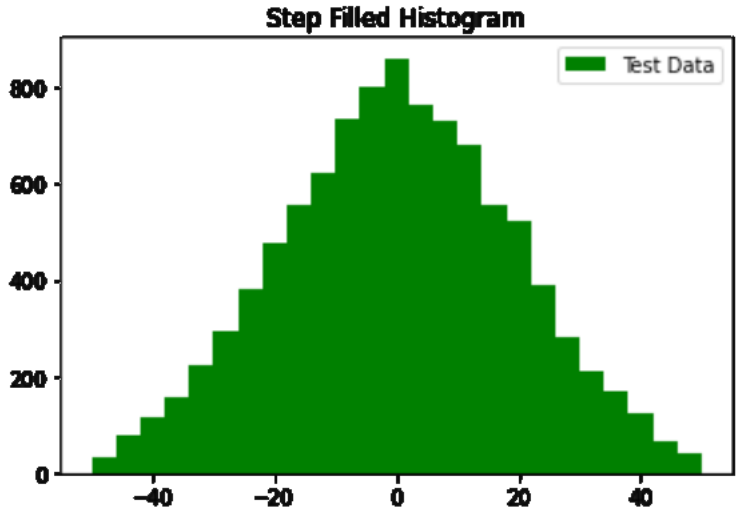
PRACTICAL NO: 3

Aim: Prepare a Histograms by taking suitable reference.

Program:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x = 20 * np.random.randn(10000)

plt.hist(x, 25, range=(-50, 50), histtype='stepfilled', align='mid', color='g', label='Test Data')
plt.legend()
plt.title('Step Filled Histogram')
plt.show()
```

Output:



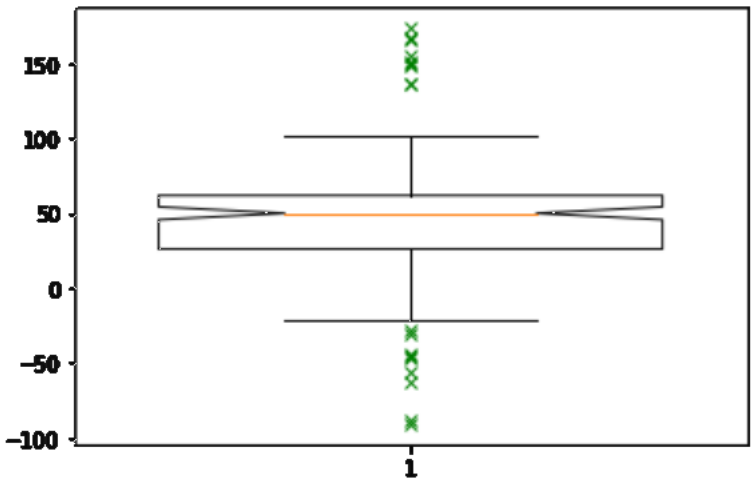
PRACTICAL NO: 4

Aim: Prepare a Boxplots by taking suitable reference.

Program:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

spread = 100 * np.random.rand(100)
center = np.ones(50) * 50
flier_high = 100 * np.random.rand(10) + 100
flier_low = -100 * np.random.rand(10)
data = np.concatenate((spread, center, flier_high, flier_low))

plt.boxplot(data, sym='gx', widths=.75, notch=True)
plt.show()
```

Output:



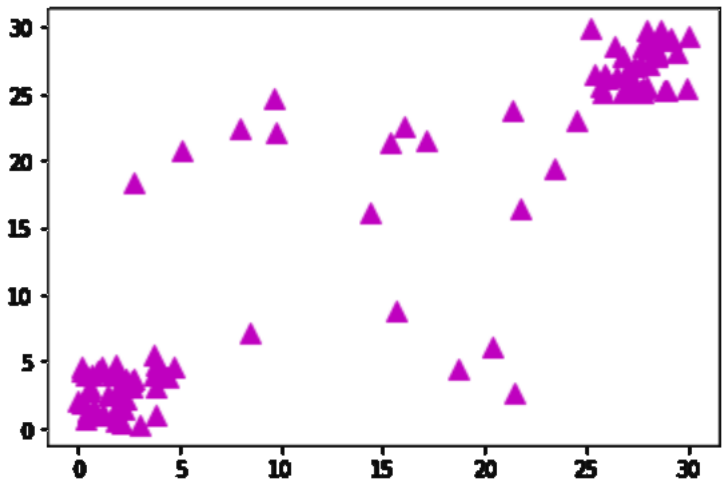
PRACTICAL NO: 5

Aim: Prepare a Scatterplots by taking suitable reference.

Program:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x1 = 5 * np.random.rand(40)
x2 = 5 * np.random.rand(40) + 25
x3 = 25 * np.random.rand(20)
x = np.concatenate((x1, x2, x3))

y1 = 5 * np.random.rand(40)
y2 = 5 * np.random.rand(40) + 25
y3 = 25 * np.random.rand(20)
y = np.concatenate((y1, y2, y3))

plt.scatter(x, y, s=[100], marker='^', c='m')
plt.show()
```

Output:



PRACTICAL NO: 6

Aim: Prepare a Time Series by taking suitable reference.

Program:

```
import pandas as pd
import matplotlib.pyplot as plt
import datetime as dt
%matplotlib inline

start_date = dt.datetime(2022, 10, 20)
end_date = dt.datetime(2022, 11, 5)
daterange = pd.date_range(start_date, end_date)
sales = (np.random.rand(len(daterange)) * 50).astype(int)
df = pd.DataFrame(sales, index=daterange, columns=['Sales'])

df.loc['Oct 4 2022':'Nov 04 2022'].plot()
plt.ylim(0,50)
plt.xlabel('Sales Date')
plt.ylabel('Sale Value')
plt.title('Plotting Time')
plt.show()
```

Output: